PYTHON PROGRAMMING

(BCC302 / BCC402/ BCC302H / BCC402H)

SYLLABUS

UNIT 2 :Python Program Flow Control Conditional blocks

if, else and else if, Simple for loops in python, For loop using ranges, string, list and dictionaries. Use of while loops in python, Loop manipulation using pass, continue, break and else. Programming using Python conditional and loop blocks

Gateway classes

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional blocks

LEC-1

By

Pragya Rajvanshi

M.Tech(CSE)

Gateway classes

If-else statement

Syntax

```
if condition:
        #block of statements
else:
      #another block of statements (else-block)
```

Write a program to find out whether a number is even or odd

```
1 x=int(input("enter the number\n"))
2 if x%2==0:
3     print("even number")
4 else:
5     print("odd number")
```

```
enter the number
5
odd number
```

Gateway classes

➢ elif statement

```
if expression 1:
    # block of statements
elif expression 2:
    # block of statements
elif expression 3:
   # block of statement
else:
    #statement
```

```
x=10
if x>10:
    print("hi")
elif x==10:
    print("hello")
```

```
hello
```

Write a program to find out the largest number among three

```
x= int(input("enter the first  number\n"))
y=int(input("enter the second number\n"))
z=int(input("enter the second number\n"))
if x>y and x>z:
    print(x,"is the largest")
elif y>x and y>z:
    print(y,"is the largest")
else:
    print(z, "is the largest")
```

```
enter the first  number
6
enter the second number
7
enter the second number
7
7 is the largest
```

Gateway classes

Write a program to simulate the calculator

```
a=int(input("enter the first number\n"))
b=int(input("enter the second number\n"))
z=input("enter any one +,-,*,%,/,// \n")
if z=="+":
    print("sum=",a+b)
elif z=="-":
    print("sub=",a-b)
elif z=="*":
    print("multiply=",a*b)
elif z=="/":
    print("subtract=",a/b)
elif z=="%":
    print("modlus=",a%b)
else:
    print("floor division",a//b)
```

```
enter the first number
5
enter the second number
5
enter any one +,-,*,%,/,//
+
sum= 10
```

Gateway classes

➢ **Nested if else**

```
if condition 1:
    if condition2:
        #nested if code
    else:
        #nested else code
else:
    #else code
```

```
enter the first number
5
enter the second number
6
enter the third number
7
7 is largest
```

Write a program to find out the largest number among three using nested –if else

```
x=int(input("enter the first number\n"))
y=int(input("enter the second number\n"))
z=int(input("enter the third number\n"))
if x>y:
    if x>z:
        print(x,"is the largest")
    else:
        print(z,"is the largest")
else:
    if y>z:
        print(y,"is the largest")
    else:
        print(z, "is largest")
```

ateway classes

Write a program to find out whether a year is leap year or not

```
x=int(input("enter the year\n"))
if x%400==0:
    print(x,"is leap year")
elif x%100==0:
    print(x,"is  not leap year")
elif x%4==0:
    print(x,"is leap year")
else:
    print(x,"is leap year")
```

```
enter the year
1600
1600 is leap year
```

Write a program to find out the roots of quadratic equation

```
from math import sqrt
a=float(input("enter the  value of a\n"))
b=float(input("enter the value of b\n"))
c=float(input("enter the value of c\n"))
d=b*b-4*a*c
if d>0:
    print("roots are real and different")
    r1=(-b+sqrt(d))/(2*a)
    r2=(-b-sqrt(d))/(2*a)
    print("root1=",r1)
    print("root2=",r2)
elif d==0:
    print("roots are real and equal")
    r1=r2=-b/(2*a)
    print("root1=",r1)
    print("root2=",r2)
else:
    print("roots are imginary")
```

Gateway Classes

```
enter the  value of a
2
enter the value of b
6.8
enter the value of c
2.5
roots are real and different
root1= -0.4193751525134304
root2= -2.9806248474865695
```

SOME BASIC QUESTIONS

Write a Python program that accepts the user's first and last name and prints them in reverse order with a space between them

```
1 fname=input("enter the firstname\n")
2 lname=input("enter the lastname\n")
3 print("hello " + lname+ " " +fname)
```

input

```
enter the firstname
pragya
enter the lastname
rajvanshi
hello rajvanshi pragya
```

Gateway classes

Write a Python program that accepts a sequence of comma-separated numbers from the user and generates a list and a tuple of those numbers.

```
values=input("enter the comma seprated value\n")
list1=values.split(",")
print(list1)
tup1=tuple(list1)
print(tup1)
```

```
enter the comma seprated value
1,2,4,5,7
['1', '2', '4', '5', '7']
('1', '2', '4', '5', '7')
```

Write a Python program to display the first and last colors and middle element from the list

```
colors = ["Red", "Green", "Blue", "Yellow", "Purple"]
print("First color:", colors[0])
print("Last color:", colors[-1])
print("Middle color",colors[5//2])
```

```
First color: Red
Last color: Purple
Middle color Blue
```

Gateway Classes

Write a Python program that accepts an integer (n) and computes the value of n+nn+nnn.

```
n=int(input("enter the number\n"))
n1=n
n2=n*n
n3=n*n*n
print(n1+n2+n3)
```

```
enter the number
2
14
```

Write a Python program to count the number 4 in a given list.

```
numbers = [1, 4, 6, 4, 8, 4, 9]
y= numbers.count(4)
print("Number of 4s in the list:", y)
```

input

```
Number of 4s in the list: 3
```

Write a Python program to calculate the number of days between two dates. *Sample dates* : (2014, 7, 2), (2014, 7, 11)

```
from datetime import date
first_date = date(2014, 7, 2)
last_date = date(2014, 7, 11)
z = last_date - first_date
print(z.days)
```

input

```
9
```

Gateway Classes

Python program that takes a string and a non-negative integer n, then returns n copies of the first two characters of the given string

```
1  string1 = "Python"
2  n = 3
3  x= string1[0:2]
4  result = x * n
5  print(result)
```

```
PyPyPy
```

Write a Python program to test whether a passed letter is a vowel or not

```
letter = input("Enter a letter: ")
if letter in 'aeiouAEIOU':
    print(letter,"is a vowel.")
else:
    print(letter,"is not vowel")
```

```
Enter a letter: A
A is a vowel.
```

Write a Python program that concatenates all elements in a list into a string and returns it.

```
elements = ['Hello', 'world', 'this']
result = ''.join(elements)
print(type(result))
print("Concatenated string:", result)
```

```
<class 'str'>
Concatenated string: Helloworldthis
```

Gateway classes

Write a Python program to sum three given integers. However, if two values are equal, the sum will be zero

```
num1 = int(input("Enter the first integer: "))
num2 = int(input("Enter the second integer: "))
num3 = int(input("Enter the third integer: "))
if num1 == num2 or num1 == num3 or num2 == num3:
    result = 0
else:
    result = num1 + num2 + num3
print("The result is:", result)
```

```
Enter the first integer: 2
Enter the second integer: 2
Enter the third integer: 3
The result is: 0
```

Write a Python program to sum two given integers. However, if the sum is between 15 and 20 it will return 20

```
num1 = int(input("Enter the first integer: "))
num2 = int(input("Enter the second integer: "))
result = num1 + num2
if 15 <= result <= 20:
    result = 20
print("The result is:", result)
```

```
Enter the first integer: 12
Enter the second integer: 3
The result is: 20
```

eway classes

Write a Python program to add two objects if both objects are integers.

```
obj1 = input("Enter the first object: ")
obj2 = input("Enter the second object: ")
if obj1.isdigit() and obj2.isdigit():
    result = int(obj1) + int(obj2)
    print("The sum is:", result)
else:
    print("Both objects are not integers.")
```

```
Enter the first object: 3
Enter the second object: 6
The sum is: 9
```

Write a Python program to sum two given integers. However, if the sum is between 15 and 20 it will return 20

```
num1 = int(input("Enter the first integer: "))
num2 = int(input("Enter the second integer: "))
result = num1 + num2
if 15 <= result <= 20:
    result = 20
print("The result is:", result)
```

```
Enter the first integer: 12
Enter the second integer: 3
The result is: 20
```

Gateway classes

Write a Python program to parse a string to float or integer

```
n = "246.2458"
print(float(n))
print(int(float(n)))
```

```
246.2458
246
```

Calculate the hypotenuse of a right angled triangle

```
from math import sqrt
print("Input lengths of shorter triangle sides:")
a = float(input("a: "))
b = float(input("b: "))
c = sqrt(a**2 + b**2)
print("The length of the hypotenuse is:", c)
```

```
Input lengths of shorter triangle sides:
a: 5
b: 4
The length of the hypotenuse is: 6.4031242374328485
```

Write a Python program to count the number of occurrences of a specific character in a string.

```
S=input("Enter a string: ")
c=input("Enter the character to count")
z=S.count(c)
print(z)
```

```
Enter a string: python
Enter the character to countp
1
```

Gateway Classes

Write a Python program to remove the first item from a specified list

```
my_list = ['apple', 'banana', 'cherry', 'date']
if my_list:
    my_list.pop(0)
print("Updated list:", my_list)
```

```
Updated list: ['apple', 'cherry', 'date']
```

Write a Python program to test whether all numbers in a list are greater than a certain number.

```
numbers = [10, 20, 30, 40, 50]
threshold = int(input("Enter the threshold number: "))
if all(num > threshold for num in numbers):
    print("YES")
else:
    print("FALSE")
```

```
Enter the threshold number: 5
YES
```

Gateway classes

Write a Python program to sum all the items in a list

```
numbers = [1,2,3,4]
result = 0
for num in numbers:
    result += num
print("The sum of all items is:", result)
```

```
The sum of all items is: 10
```

Write a Python program to multiply all the items in a list.

```
1 numbers = [1,2,3,4]
2 result = 1
3 for num in numbers:
4     result *= num
5 print("The product of all items is:", result)
```

input

```
The product of all items is: 24
```

Write a Python program to get the largest number from a list

```
1 numbers = [10, 25, 47, 3, 99, 56]
2 largest = numbers[0]
3 for num in numbers:
4     if num > largest:
5         largest = num
6 print("The largest number is:", largest)
```

```
The largest number is: 99
```

Gateway classes

Write a Python program to count the number of strings from a given list of strings. The string length is 2 or more and the first and last characters are the same

```
strings = ['abc', 'xyz', 'aba', '1221', 'a']
count = 0
for s in strings:
    if len(s) >= 2 and s[0] == s[-1]:
        count += 1
print( count)
```

```
2
```

Write a Python program to get the smallest number from a list

```
1  numbers = [10, 25, 47, 3, 99, 56]
2  smallest = numbers[0]
3  for num in numbers:
4      if num < smallest:
5          smallest = num
6  print("The smallest number is:", smallest)
```

```
The smallest number is: 3
```

Gateway classes

PYTHON PROGRAMMING

(BCC302 / BCC402/ BCC302H / BCC402H)

SYLLABUS

UNIT 2 :Python Program Flow Control Conditional blocks

if, else and else if, Simple for loops in python, For loop using ranges, string, list and dictionaries. Use of while loops in python, Loop manipulation using pass, continue, break and else. Programming using Python conditional and loop blocks

Gateway classes

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional blocks

LEC-2

By

Pragya Rajvanshi
M.Tech(CSE)

Gateway classes

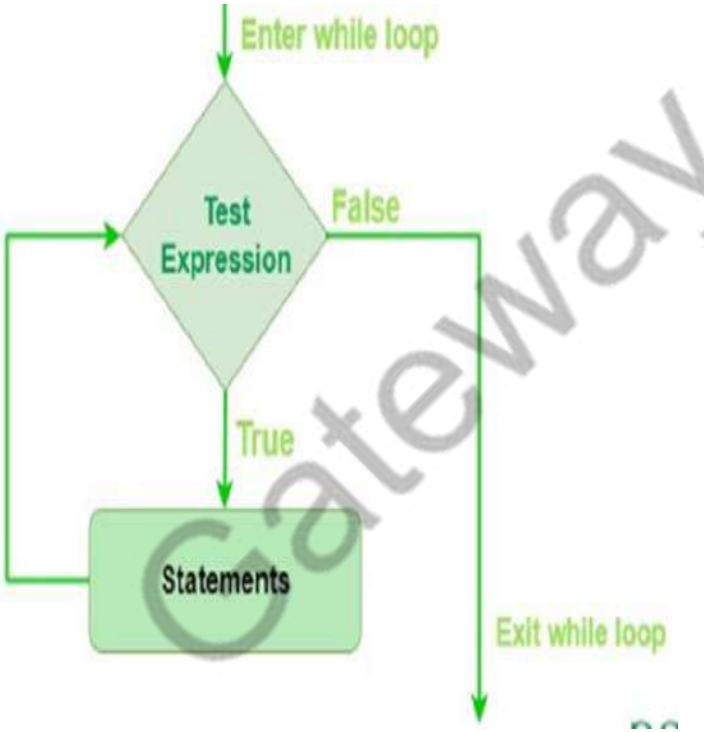
LOOPS

- Loops is used to execute a block of statements repeatedly until a given condition is satisfied.

1 while loop

Syntax

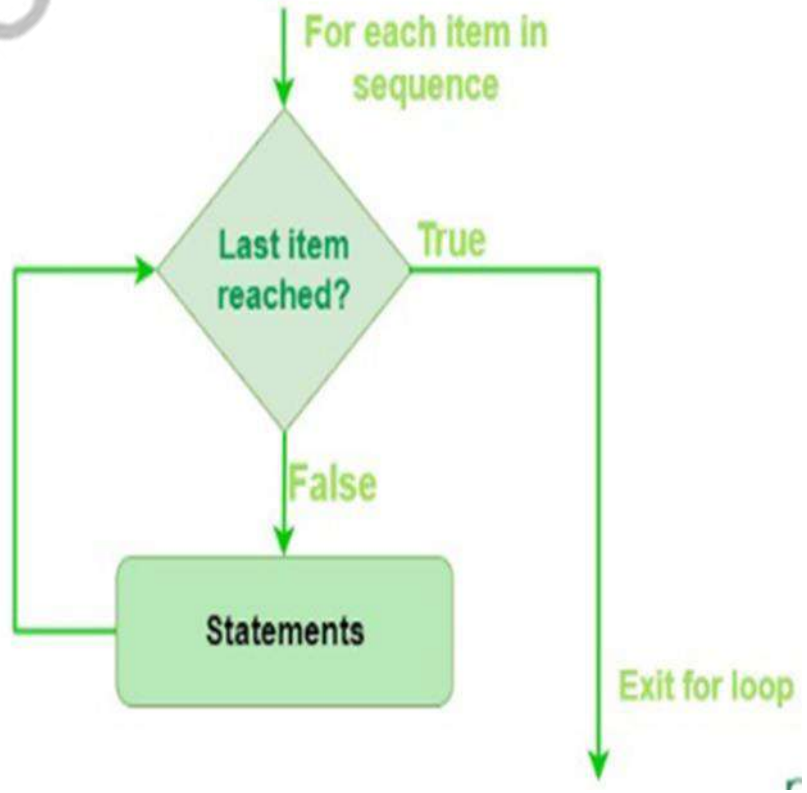
```
while expression:
    statement(s)
```



2 For loop

Syntax

```
for var in iterable:
    # statements
```



Gateway classes

➢ Range ()

➢ The Python **range() function** returns a sequence of numbers, in a given range.

1. range (stop)

```
1 for i in range(6):
2     print(i, end=" ")
3
```

```
0 1 2 3 4 5
```

2.range (start, stop)

```
3 for i in range(5, 20):
4     print(i, end=" ")
5
```

```
5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

3.range (start, stop, step)

```
1 for i in range(0, 30, 4):
2     print(i, end=" ")
3
```

```
0 4 8 12 16 20 24 28
```

Gateway classes

Write a for loop that prints number from 0 to 57 using range function

```
1 for i in range(58):
2     print(i)
3
```

write a program to check whether a number is Armstrong or not

```
1  n=int(input("enter the number"))
2  sum = 0
3  num=n
4  while n> 0:
5      digit = n% 10
6      sum += digit ** 3
7      n //= 10
8  if num==sum:
9      print("armstromg")
10 else:
11     print("not armstrong")
12
```

```
enter the number153
armstromg
```

Gateway classes

write a program to find out the whether a number is palindrome or not

```
n=int(input("enter n digit number\n"))
num=n
rev=0
while n>0:
    d=n%10
    rev=rev*10+d
    n=n//10
print("reverse",rev)
if rev== num:
    print("palindrome")
else:
    print("not palindrome")
```

```
enter n digit number
121
reverse 121
palindrome
```

write a program to find out the fibonnaci series upto n term

```

nterms = int(input("How many terms? "))
n1, n2 = 0, 1
count = 0
if nterms <= 0:
   print("Please enter a positive integer")
elif nterms == 1:
   print(n1)
else:
    print("Fibonacci sequence:")
    print(n1,n2)
    for i in range(3,nterms+1):
       nth = n1 + n2
       print(nth)
       n1 = n2
       n2 = nth

```

```
How many terms? 3
Fibonacci sequence:
0 1
1
```

gateway classes

Write a program to find out the factorial of a number

```
num=int(input("enter the number\n"))
factorial = 1
if num < 0:
   print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
else:
   for i in range(1,num + 1):
       factorial = factorial*i
   print("The factorial of",num,"is",factorial)

```

input

```
enter the number
4
The factorial of 4 is 24
```

write a program to check whether a number is prime or not

```
num = int(input("Enter a number: "))
flag = False
if num == 1:
    print(num, "is not a prime number")

elif num>1:
    for i in range(2, num):
        if (num % i) == 0:
            flag = True
            break
    if flag:
        print(num, "is not a prime number")
    else:
        print(num, "is a prime number")
else:
    print("enter positive number")
```

input

```
Enter a number: 7
7 is a prime number
```

Gateway classes

write a program to print the reverse of a number

```
1 n=int(input("enter n digit number\n"))
2 num=n
3 rev=0
4 while n>0:
5     d=n%10
6     rev=rev*10+d
7     n=n//10
8 print("reverse",rev)
9
```

```
enter n digit number
1214
reverse 4121
```

What will be the output?

```
1 i=0
2 while i<3:
3     print(i)
4     i+=1
5 else:
6     print(0)
```

```
0
1
2
0
```

teeway classes

write a program to convert time from 12 hour to 24 hour format

```
str1="12:05:45 AM"
if str1[-2:]=="AM" and str1[:2]=="12":
    print("00"+str1[2:-2])
elif str1[-2:]=="AM":
    print(str1[:-2])
elif str1[-2:]=="PM" and str1[:2]=="12":
    Print(str1[:-2])
else:
    print(str(int(str1[:2])+12)+str1[2:8])
```

```
00:05:45
```

Gateway classes

break statement

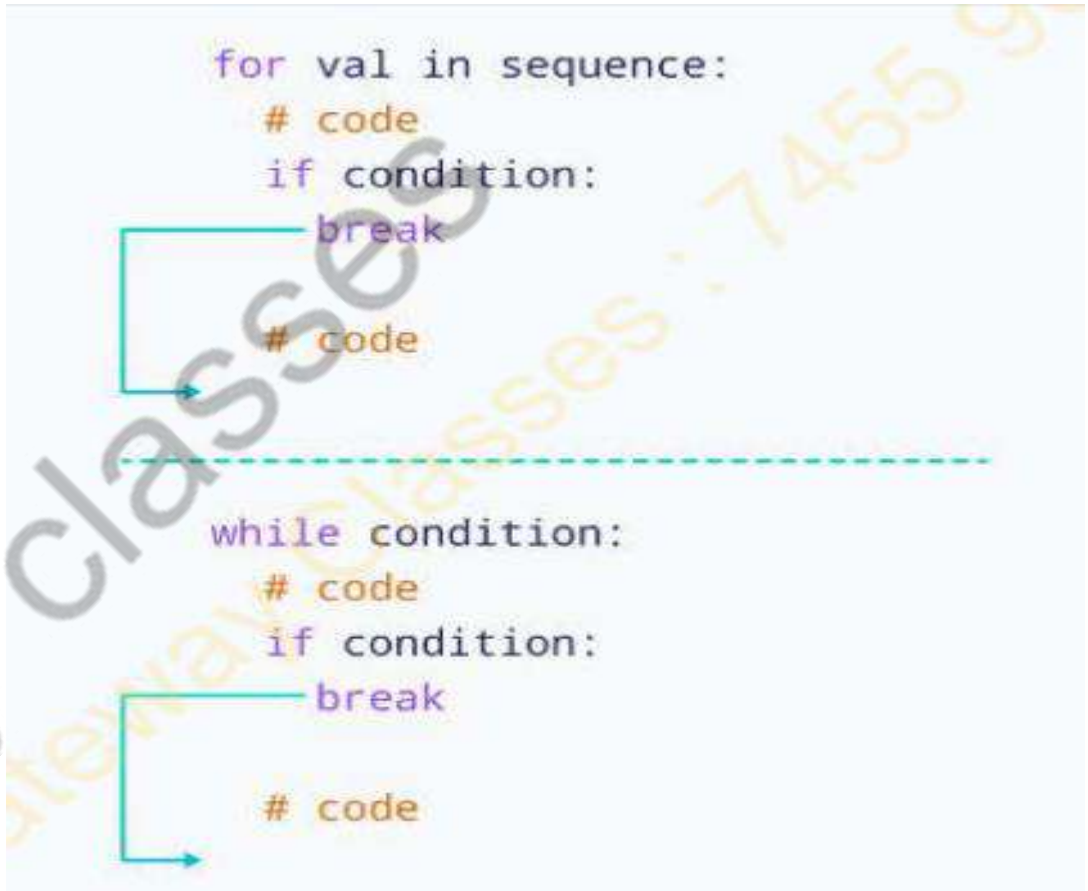
The break statement is used to terminate the loop immediately when it is encountered.

The syntax of the break statement is:

break

```
1 for i in range(5):
2     if i == 3:
3         break
4     print(i)
```

```
0
1
2
```



Gateway classes

continue statement

The continue statement is used to skip the current iteration of the loop and the control flow of the program goes to the next iteration.

The syntax Continue;

```
for i in range(5):
    if i == 3:
        continue
    print(i)
```

```
0
1
2
4
```

```
for val in sequence:
    # code
    if condition:
        continue

    # code
```

```
while condition:
    # code
    if condition:
        continue

    # code
```

Gateway classes : 7455 96

Continue Example

main.py

```
num = 0
while num < 10:
    num += 1
    if (num % 2) == 0:
        continue
    print(num)
```

```
1
3
5
7
9
```

Break example

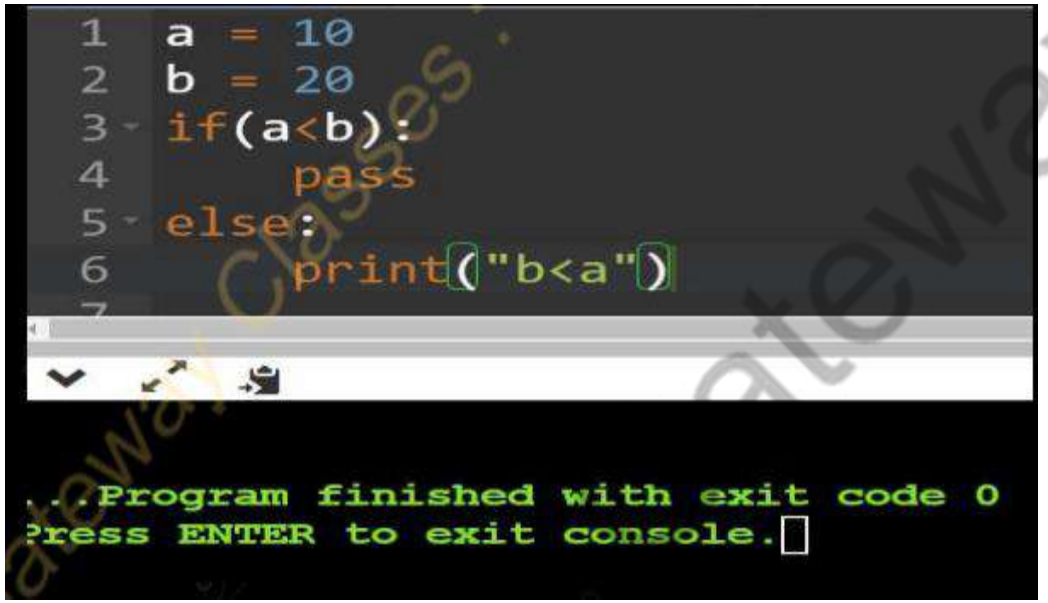
```
i = 1
while i <= 10:
    print('6 * ',(i),'=',6 * i)
    if i >= 5:
        break
    i = i + 1
```

```
6 *  1 = 6
6 *  2 = 12
6 *  3 = 18
6 *  4 = 24
6 *  5 = 30
```

Gateway Classes : 7455 961

Pass example

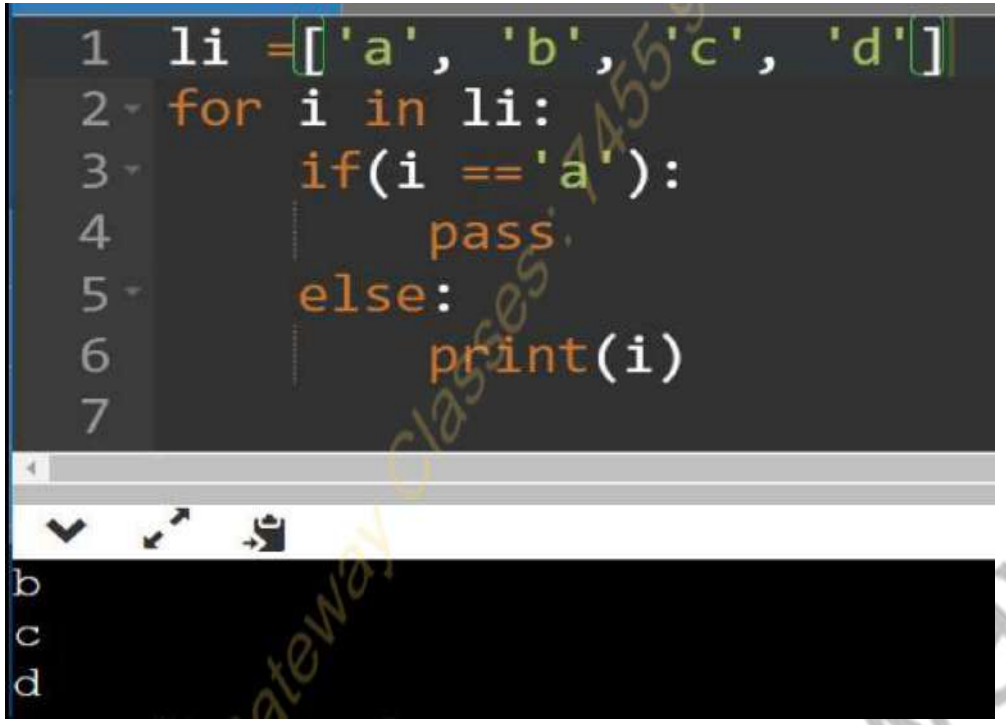
- python pass statement is a null statement. But the difference between pass and comment is that comment is ignored by the interpreter whereas pass is not ignored.
- SYNTAX pass



What is pass statement in Python?

- When the user does not know what code to write, So user simply places a pass at that line.
- Sometimes, the pass is used when the user doesn't want any code to execute.
- So users can simply place a pass where empty code is not allowed, like in loops, function definitions, class definitions, or in if statements. So using a pass statement user avoids this error.

Gateway Classes



Gateway classes

Write a program to print the prime number in range

main.py

```
1  lower = int(input("enter the lower range"))
2  upper = int(input ("enter the upper range"))
3  print("Prime numbers between", lower, "and", upper, "are:")
4  for num in range(lower, upper + 1):
5      if num > 1:
6          for i in range(2, num):
7              if (num % i) == 0:
8                  break
9          else:
10             print(num)
11
```

input

```
enter the lower range2
enter the upper range7
Prime numbers between 2 and 7 are:
2
3
5
7
```

way classes

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional blocks

LEC-3

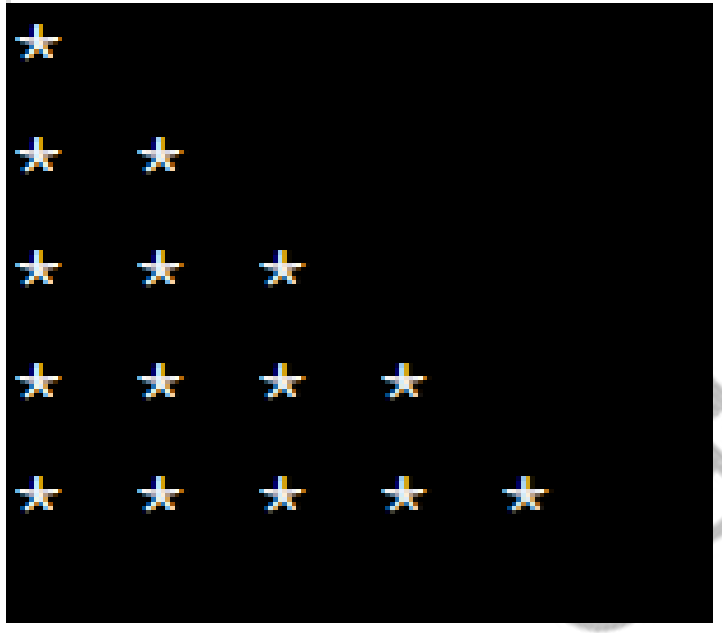
By

Pragya rajvanshi

Gateway classes

Pattern printing-1

```
rows = 5
for i in range(1, rows+1):
    print('* ' * i)
```



Pattern printing-2

```
rows = 5
for i in range(rows,0,-1):
    print('* ' * i)
```



Gateway classes

Pattern printing-3

```
rows = 5
for i in range(1, rows + 1):
    print(' '.join(str(x) for x in range(1, i + 1)))
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Pattern printing-4

```
rows = 5
for i in range(rows, 0, -1):
    print(' '.join(str(x) for x in range(1, i + 1)))
```

```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

Gateway classes

Pattern printing-5

```
rows = 5
for i in range(1, rows + 1):
    print(' '.join(chr(64 + i) for _ in range(i)))
```

```
A
B B
C C C
D D D D
E E E E E
```

Pattern printing-6

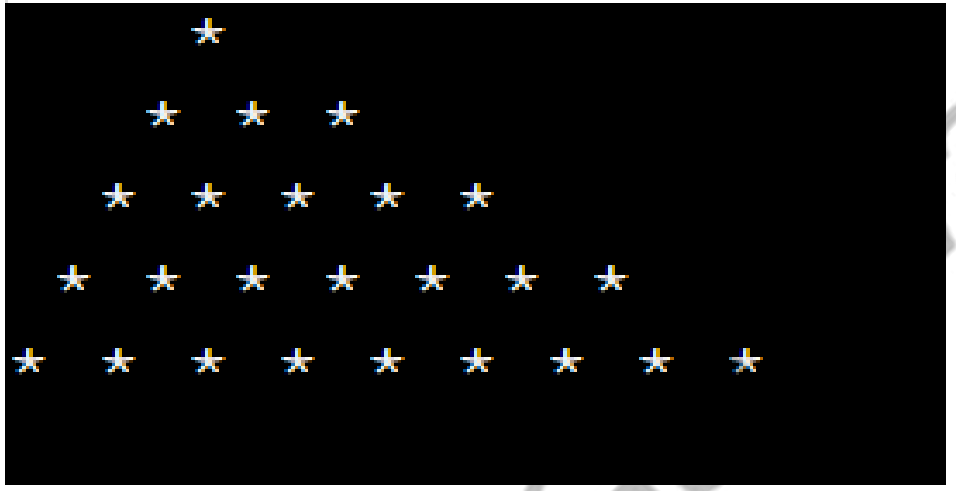
```
rows = 5
for i in range(rows, 0, -1):
    print(' '.join(chr(64 + i) for _ in range(i)))
```

```
E E E E E
D D D D
C C C
B B
A
```

Oneway classes

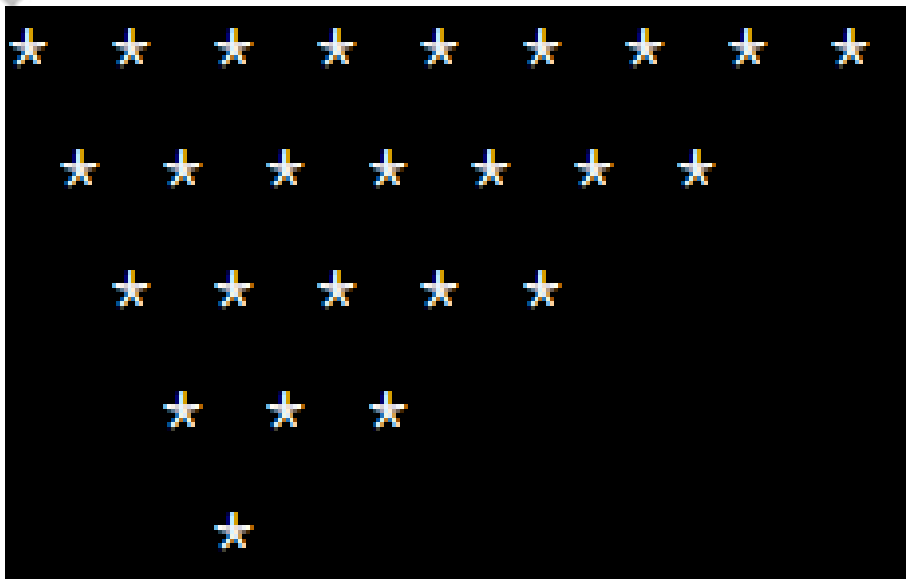
Pattern printing-7

```
rows = 5
for i in range(1, rows + 1):
    print(' ' * (rows - i) + '* ' * (2 * i - 1))
```



Pattern printing-8

```
rows = 5
for i in range(rows,0,-1):
    print(' ' * (rows - i) + '* ' * (2 * i - 1))
```



Gurukay classes

1. Write a Python script to add a key to a dictionary

```
my_dict = {0: 10, 1: 20}
my_dict[2] = 30
print("Updated dictionary:", my_dict)
```

```
Updated dictionary: {0: 10, 1: 20, 2: 30}
```

2. Write a Python script to concatenate/merge the following dictionaries to create a new one.

```
dict1 = {1: 10, 2: 20}
dict2 = {3: 30, 4: 40}
dict3 = {5: 50, 6: 60}
new_dict = {}
for d in (dict1, dict2, dict3):
    new_dict.update(d)
print("Concatenated dictionary:", new_dict)
```

```
Concatenated dictionary: {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
```

Gateway classes

3. Write a Python script to check whether a given key already exists in a dictionary.

```
my_dict = {1: 10, 2: 20, 3: 30}
key_to_check = 2
if key_to_check in my_dict:
    print("yes")
else:
    print("no")
```

```
yes
```

4.Write a Python program to iterate over dictionaries using for loops

```
dict1 = {'apple': 10, 'banana': 20, 'cherry': 30}
print("Keys:")
for key in dict1:
    print(key)
print("\nValues:")
for value in dict1.values():
    print(value)
print("\nKey-Value Pairs:")
for key, value in dict1.items():
    print(f"{key}: {value}")
```

Gateway Classes

```
Keys:
apple
banana
cherry

Values:
10
20
30

Key-Value Pairs:
apple: 10
banana: 20
cherry: 30
```

5. Write a Python script to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).

```
n = int(input("Enter a number: "))
squared_dict = {x: x*x for x in range(1, n+1)}
print("Generated dictionary:", squared_dict)
```

```
Enter a number: 5
Generated dictionary: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

6.Write a Python script to print a dictionary where the keys are numbers between 1 and 15 (both included) and the values are the square of the keys.

```
squared_dict = {x: x*x for x in range(1, 16)}
print("Generated dictionary:", squared_dict)
```

```
Generated dictionary: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81
, 10: 100, 11: 121, 12: 144, 13: 169, 14: 196, 15: 225}
```

Gateway Classes

7. Write a Python program to sum all the items in a dictionary

```
my_dict = {'a': 10, 'b': 20, 'c': 30}
total_sum = sum(my_dict.values())
print("Total sum of values:", total_sum)
```

```
Total sum of values: 60
```

8.Write a Python program to Multiply all the items in a dictionary

```
my_dict = {'a': 2, 'b': 3, 'c': 4}
result = 1
for value in my_dict.values():
    result *= value
print("Product of all values:", result)
```

```
Product of all values: 24
```

9. Write a Python program to remove a key from a dictionary.

```
my_dict = {'a': 1, 'b': 2, 'c': 3}
key_to_remove = 'b'
if key_to_remove in my_dict:
    del my_dict[key_to_remove]
print("Updated dictionary:", my_dict)
```

```
Updated dictionary: {'a': 1, 'c': 3}
```

Gateway Classes

10.Write a Python program to map two lists into a dictionary.

```
keys = ['a', 'b', 'c']
values = [1, 2, 3]
mapped_dict = dict(zip(keys, values))
print("Mapped dictionary:", mapped_dict)
```

11 Write a Python program to sort a given dictionary by key.

```
my_dict = {'banana': 3, 'grapes': 1, 'cherry': 2}
sorted_dict = dict(sorted(my_dict.items()))
print("Dictionary sorted by key:", sorted_dict)
```

```
Dictionary sorted by key: {'banana': 3, 'cherry': 2, 'grapes': 1}
```

12. Write a Python program to get the maximum and minimum values of a dictionary.

```
my_dict = {'a': 10, 'b': 20, 'c': 30, 'd': 5}
max_value = max(my_dict.values())
min_value = min(my_dict.values())
print("Maximum value:", max_value)
print("Minimum value:", min_value)
```

```
Maximum value: 30
Minimum value: 5
```

Gateway classes

13. Write a Python program to remove duplicates from the dictionary.

```
my_dict = {'a': 10, 'b': 20, 'c': 10, 'd': 30, 'e': 20}
unique_dict = {}
for key, value in my_dict.items():
    if value not in unique_dict.values():
        unique_dict[key] = value
print("Dictionary without duplicates:", unique_dict)
```

```
Dictionary without duplicates: {'a': 10, 'b': 20, 'd': 30}
```

14 Write a Python program to check if a dictionary is empty or not.

```
my_dict = {}
if my_dict:
    print("The dictionary is  not empty.")
else:
    print("The dictionary is  empty.")
```

```
The dictionary is  empty.
```

15. Write a Python program to combine two dictionary by adding values for common keys.

d1 = {'a': 100, 'b': 200, 'c':300}

d2 = {'a': 300, 'b': 200, 'd':400}

Sample output: Counter({'a': 400, 'b': 400, 'd': 400, 'c': 300})

```
from collections import Counter
d1 = {'a': 100, 'b': 200, 'c': 300}
d2 = {'a': 300, 'b': 200, 'd': 400}
combined_dict = Counter(d1) + Counter(d2)
print("Combined dictionary:", combined_dict)
```

Gateway classes

16. Write a Python program to print all distinct values in a dictionary.

Sample Data : [{"V":"S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII":"S005"}, {"V":"S009"},{"VIII":"S007"}]

Expected Output : Unique Values: {'S005', 'S002', 'S007', 'S001', 'S009'}

```
data = [{"V": "S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"},
        {"VII": "S005"}, {"V": "S009"}, {"VIII": "S007"}]
distinct_values = set()
for dictionary in data:
    distinct_values.update(dictionary.values())
print("Distinct values:", distinct_values)
```

```
Distinct values: {'S005', 'S009', 'S007', 'S001', 'S002'}
```

Gateway classes

17. Write a Python program to create and display all combinations of letters, selecting each letter from a different key in a dictionary.
Sample data : {'1':['a','b'], '2':['c','d']}

```
from itertools import product
data = {'1': ['a', 'b'], '2': ['c', 'd']}
combinations = product(*data.values())
for combo in combinations:
    print(''.join(combo))
```

```
ac
ad
bc
bd
```

18. Write a Python program to create a dictionary from a string.
Note: Track the count of the letters from the string

```
sample_string = 'pythonpro'
char_count = {}
for char in sample_string:
    if char in char_count:
        char_count[char] += 1
    else:
        char_count[char] = 1
print(char_count)
```

```
{'p': 2, 'y': 1, 't': 1, 'h': 1, 'o': 2, 'n': 1, 'r': 1}
```

Gateway classes

19. Write a Python program to remove spaces from dictionary keys

```
original_dict = {"first name": "John", "last name": "Doe", "age": 25}
new_dict = {key.replace(" ", ""): value for key, value in original_dict.items()}
print("Dictionary with no spaces in keys:", new_dict)

```

input

```
Dictionary with no spaces in keys: {'firstname': 'John', 'lastname': 'Doe', 'age': 25}
```

20. Write a Python program to check if multiple keys exist in a dictionary.

```
my_dict = {'a': 1, 'b': 2, 'c': 3}
keys_to_check = ['a', 'b']
all_keys_exist = all(key in my_dict for key in keys_to_check)
if all_keys_exist:
    print("All keys exist in the dictionary.")
else:
    print("Not all keys exist in the dictionary.")
```

```
All keys exist in the dictionary.
```

Gateway Classes

21. Write a Python program to get the key, value and item in a dictionary

```
my_dict = {'a': 100, 'b': 200, 'c': 300}
for key, value in my_dict.items():
    print(f"Key: {key}, Value: {value}, Item: ({key}, {value})")
```

```
Key: a, Value: 100, Item: (a, 100
Key: b, Value: 200, Item: (b, 200
Key: c, Value: 300, Item: (c, 300
```

22 Write a Python program to print a dictionary line by line.

```
my_dict = {'a': 100, 'b': 200, 'c': 300}
for key, value in my_dict.items():
    print(f"{key}: {value}")
```

```
a:  100
b:  200
c:  300
```

23. Write a Python program to replace dictionary values with their sums.

```
my_dict = {'a': [1, 2, 3], 'b': [4, 5], 'c': [6, 7, 8]}
for key in my_dict:
    my_dict[key] = sum(my_dict[key])
print(my_dict)
```

```
{'a': 6, 'b': 9, 'c': 21}
```

Gateway classes

Dictionary

24.Write a Python program to create a dictionary of keys x, y, and z where each key has as value a list from 11-20, 21-30, and 31-40 respectively. Access the fifth value of each key from the dictionary.

{'x': [11, 12, 13, 14, 15, 16, 17, 18, 19],

'y': [21, 22, 23, 24, 25, 26, 27, 28, 29],

'z': [31, 32, 33, 34, 35, 36, 37, 38, 39]}

15

25

35

x has value [11, 12, 13, 14, 15, 16, 17, 18, 19]

y has value [21, 22, 23, 24, 25, 26, 27, 28, 29]

z has value [31, 32, 33, 34, 35, 36, 37, 38, 39]

```
my_dict = {
    'x': list(range(11, 20)),
    'y': list(range(21, 30)),
    'z': list(range(31, 40))
}
# Accessing the fifth value from each key
print(my_dict['x'][4])  # 15
print(my_dict['y'][4])  # 25
print(my_dict['z'][4])  # 35
# Printing the entire dictionary with the lists
for key, value in my_dict.items():
    print(f"{key} has value {value}")
```

Gateway classes

25. Write a Python program to match key values in two dictionaries.

Sample dictionary: {'key1': 1, 'key2': 3, 'key3': 2}, {'key1': 1, 'key2': 2}

Expected output: key1: 1 is present in both x and y

```
# Sample dictionaries
dict1 = {'key1': 1, 'key2': 3, 'key3': 2}
dict2 = {'key1': 1, 'key2': 2}

# Compare and find matching keys with the same values
for key in dict1:
    if key in dict2 and dict1[key] == dict2[key]:
        print(f"{key}: {dict1[key]} is present in both x and y")
```

```
key1: 1 is present in both x and y
```

Gateway classes

26.Write a Python program to find the key of the maximum value in a dictionary.

Sample Output:

Original dictionary elements:

{'Theodore': 19, 'Roxanne': 22, 'Mathew': 21, 'Betty': 20}

Finds the key of the maximum and minimum value of the said dictionary:

('Roxanne', 'Theodore')

```
my_dict = {'Theodore': 19, 'Roxanne': 22, 'Mathew': 21, 'Betty': 20}
max_key = max(my_dict, key=my_dict.get)
min_key = min(my_dict, key=my_dict.get)
print("Original dictionary elements:")
print(my_dict)
print((max_key, min_key))
```

```
Original dictionary elements:
{'Theodore': 19, 'Roxanne': 22, 'Mathew': 21, 'Betty': 20}
('Roxanne', 'Theodore')
```

Gateway Classes

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional blocks

By

Pragya rajvanshi

LEC-4(list)

Gateway classes

1. Write a Python program to remove duplicates from a list.

```
1 a = [1, 2, 2, 3, 4, 4, 5]
2 unique = list(set(a))
3 print("duplicates removed:", unique)
```

```
duplicates removed: [1, 2, 3, 4, 5]
```

2. Write a Python program to check if a list is empty or not.

```
1 a= []
2 if not a:
3     print("The list is empty.")
4 else:
5     print("The list is not empty.")
```

```
The list is empty.
```

3. Write a Python program to clone or copy a list

```
1 list = [1, 2, 3, 4, 5]
2 copied_list = list.copy()
3 print("Original list:", list)
4 print("Copied list:", copied_list)
```

```
Original list: [1, 2, 3, 4, 5]
Copied list: [1, 2, 3, 4, 5]
```

4.Write a Python program to find the list of words that are longer than n from a given list of words.

```
1 words = ["apple", "banana", "cherry", "date", "fig", "grape"]
2 n = 5
3 long_words = [x for x in words if len(x) > n]
4 print("Words longer than", n, "characters:", long_words)
```

```
input
Words longer than 5 characters: ['banana', 'cherry']
```

Gateway classes

5. Write a Python program to print a specified list after removing the 0th, 4th and 5th elements. Sample List : ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow'] Expected Output : ['Green', 'White', 'Black']

```
sample_list = ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow']
result = [sample_list[i] for i in range(len(sample_list)) if i not in (0, 4, 5)]
print("Expected Output:", result)
```

```
Expected Output: ['Green', 'White', 'Black']
```

6. Write a Python program to print the numbers of a specified list after removing even numbers from it

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
odd_numbers = [num for num in numbers if num % 2 != 0]
print("List after removing even numbers:", odd_numbers)
```

Gateway Classes

7. Write a Python program to shuffle and print a specified list

```
1  import random
2  sample_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
3  random.shuffle(sample_list)
4  print("Shuffled list:", sample_list)
```

```
Shuffled list: [8, 5, 7, 9, 1, 6, 10, 3, 2, 4]
```

8. Whose number square is less than equal to 30 range is taken between 1 to 30

```
square_numbers = [i**2 for i in range(1, 31) if i**2 <= 30]
print(square_numbers)
first_five = square_numbers[:5]
last_five = square_numbers[-5:]
print("First 5 square numbers:", first_five)
print("Last 5 square numbers:", last_five)
```

```
[1, 4, 9, 16, 25]
First 5 square numbers: [1, 4, 9, 16, 25]
Last 5 square numbers: [1, 4, 9, 16, 25]
```

Gateway classes

9. Write a Python program to check if each number is prime in a given list of numbers. Return True if all numbers are prime otherwise False.

```
numbers = [2, 3, 5, 7, 11]
all_prime = True
for num in numbers:
    if num < 2:
        all_prime = False
        break
    for i in range(2, num):
        if num % i == 0:
            all_prime = False
            break
print("All numbers are prime:", all_prime)
```

```
All numbers are prime: True
```

10.Write a Python program to calculate the difference between the two lists.

```
list1 = [1, 2, 3, 4, 5]
list2 = [4, 5, 6, 7, 8]
difference = [item for item in list1 if item not in list2]
print("Difference between the two lists:", difference)
```

```
Difference between the two lists: [1, 2, 3]
```

tewayclasses

11. Write a Python program to access the index of a list.

```
nums = [5, 15, 35, 8, 98]
for num_index, num_val in enumerate(nums):
    print(num_index, num_val)
```

12. Write a Python program to convert a list of characters into a string.

```
char_list = ['H', 'e', ' ', 'W', 'o']
result_string = ''.join(char_list)
print("Converted string:", result_string)
```

```
Converted string: He Wo
```

13. Write a Python program to find the index of an item in a specified list.

```
sample_list = ['apple', 'banana', 'cherry', 'date', 'fig']
item_to_find = 'cherry'
if item_to_find in sample_list:
    index = sample_list.index(item_to_find)
    print(f"The index of '{item_to_find}' is: {index}")
else:
    print(f"'{item_to_find}' is not in the list.")
```

```
The index of 'cherry' is: 2
```

14. Write a Python program to append a list to the second list.

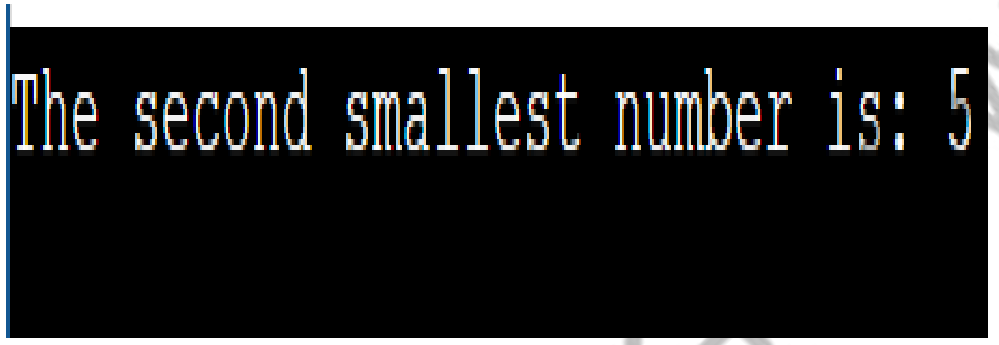
```
list1 = [1, 2, 3]
list2 = [4, 5, 6]
list2.extend(list1)
print("Modified second list:", list2)
```

```
Modified second list: [4, 5, 6, 1, 2, 3]
```

Gateway classes

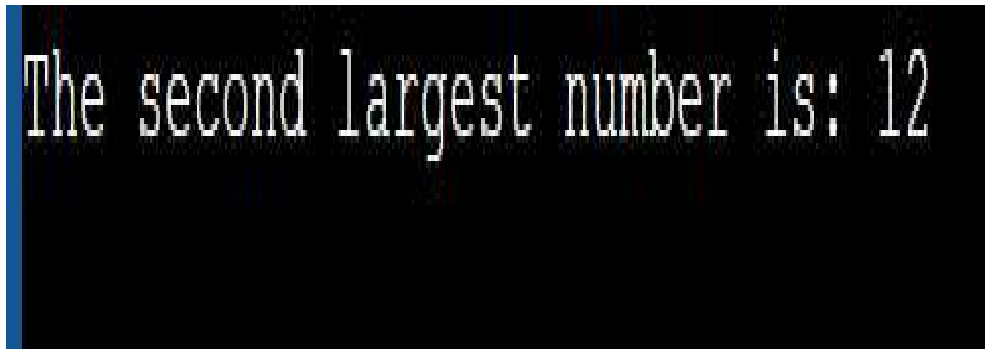
15. Python program to find the second smallest number in a list:

```
numbers = [5, 15, 3, 8, 9, 12]
unique_numbers = list(set(numbers))
unique_numbers.sort()
if len(unique_numbers) >= 2:
    second_smallest = unique_numbers[1]
    print("The second smallest number is:", second_smallest)
else:
    print("There is no second smallest number.")
```



16 . Python program to find the second largest number in a list:

```
numbers = [5, 15, 3, 8, 9, 12]
unique_numbers = list(set(numbers))
unique_numbers.sort(reverse=True)
if len(unique_numbers) >= 2:
    second_largest = unique_numbers[1]
    print("The second largest number is:", second_largest)
else:
    print("There is no second largest number.")
```



Gateway classes

17. Write a Python program to get unique values from a list.

```
numbers = [1, 2, 3, 2, 4, 5, 1, 6, 4]
unique_values = list(set(numbers))
print("Unique values in the list:", unique_values)
```

```
The second largest number is: 12
```

18.Write a Python program to get the frequency of elements in a list

```
elements = ['apple', 'banana', 'apple', 'orange', 'banana', 'apple']
frequency = {}
for item in elements:
    if item in frequency:
        frequency[item] += 1
    else:
        frequency[item] = 1
print("Frequency of elements in the list:")
for item, count in frequency.items():
    print(f"{item}: {count}")
```

```
Frequency of elements in the list:
apple: 3
banana: 2
orange: 1
```

Gateway class

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional Blocks

By

Pragya Rajvanshi

LEC-5(list)

Gateway classes

1. Write a Python program to count the number of elements in a list within a specified range.

```
numbers = [5, 15, 3, 8, 9, 12, 20, 25, 30]
lower_bound = 10
upper_bound = 15
count_in_range = sum(1 for num in numbers if lower_bound <= num <= upper_bound)
print(f"Number of elements in the list within the range {lower_bound} to {upper_bound}: {count_in_range}")
```

```
Number of elements in the list within the range 10 to 15: 2
```

2. Write a Python program to check whether a list contains a sub list.

```
main_list = [1,4, 5, 6, 7, 8, 9]
sub_list = [4, 5, 6]
found = False
for i in range(len(main_list) - len(sub_list) + 1):
    if main_list[i:i + len(sub_list)] == sub_list:
        found = True
        break
if found:
    print("The main list contains the sublist.")
else:
    print("The main list does not contain the sublist.")
```

```
The main list contains the sublist.
```

Gateway Classes

3. Write a Python program to find common items in two lists

```
list1 = [1, 2, 3, 4, 5]
list2 = [4, 5, 6, 7, 8]
common_items = []
for item in list1:
    if item in list2:
        common_items.append(item)
print("Common items:", common_items)
```

```
Common items: [4, 5]
```

4. Write a Python program to convert a list of multiple integers into a single integer.

```
list1 = [1, 23, 456, 7, 89]
result_string = ""
for num in list1:
    result_string += str(num)
single_integer = int(result_string)
print("The single integer is:", single_integer)
```

```
The single integer is: 123456789
```

Gateway classes

5. Write a Python program to split a list based on the first character of a word.

```
words = ['apple', 'banana', 'apricot', 'blueberry', 'cherry', 'date', 'fig', 'grape']
split_dict = {}
for word in words:
    first_char = word[0]
    if first_char not in split_dict:
        split_dict[first_char] = []
    split_dict[first_char].append(word)
for char, word_list in split_dict.items():
    print(f"Words starting with '{char}': {word_list}")
```

```
Words starting with 'a': ['apple', 'apricot']
Words starting with 'b': ['banana', 'blueberry']
Words starting with 'c': ['cherry']
Words starting with 'd': ['date']
Words starting with 'f': ['fig']
Words starting with 'g': ['grape']
```

6. Write a Python program to create multiple lists.

```
list1 = [1, 2, 3]
list2 = ['a', 'b', 'c']
list3 = [True, False, True]
list4 = [3.14, 2.71, 1.41]
multiple_lists = [list1, list2, list3, list4]
print("List 1:", list1)
print("List 2:", list2)
print("List 3:", list3)
print("List 4:", list4)
print("Multiple Lists:", multiple_lists)
```

```
List 1: [1, 2, 3]
List 2: ['a', 'b', 'c']
List 3: [True, False, True]
List 4: [3.14, 2.71, 1.41]
Multiple Lists: [[1, 2, 3], ['a', 'b', 'c'], [True, False, True], [3.14, 2.71, 1.41]]
```

Gateway classes

7. Write a Python program to split a list into different variables.

```
my_list = [10, 20, 30, 40, 50]
var1, var2, var3, var4, var5 = my_list
print("Variable 1:", var1)
print("Variable 2:", var2)
print("Variable 3:", var3)
print("Variable 4:", var4)
print("Variable 5:", var5)
```

```
Variable 1: 10
Variable 2: 20
Variable 3: 30
Variable 4: 40
Variable 5: 50
```

8. Write a Python program to insert an element before each element of a list.

```
original_list = [10, 20, 30, 40, 50]
element_to_insert = 'X'
modified_list = []
for item in original_list:
    modified_list.append(element_to_insert)
    modified_list.append(item)
print("Modified List:", modified_list)
```

```
Modified List: ['X', 10, 'X', 20, 'X', 30, 'X', 40, 'X', 50]
```

Gateway classes

9. Write a Python program to split a list every Nth element.

Sample list: ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n']

Expected Output: [['a', 'd', 'g', 'j', 'm'], ['b', 'e', 'h', 'k', 'n'], ['c', 'f', 'i', 'l']]

```
sample_list = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n']
N = 3
split_lists = []
for i in range(N):
    sublist = sample_list[i::N]
    split_lists.append(sublist)
print("Resulting Split Lists:")
print(split_lists)
```

```
Resulting Split Lists:
[['a', 'd', 'g', 'j', 'm'], ['b', 'e', 'h', 'k', 'n'], ['c', 'f', 'i', 'l']]
```

Gateway classes

10. Write a Python program to concatenate elements of a list.

```
string_list = ['Hello', 'World', 'from']
concatenated_string = ' '.join(string_list)
print("Concatenated String:", concatenated_string)
```

```
Concatenated String: Hello World from
```

11.. Write a Python program to convert a string to a list.

```
input_string = "Hello, World!"
result = list(input_string)
print(result)
```

```
['H', 'e', 'l', 'l', 'o', ',', ' ', 'W', 'o', 'r', 'l', 'd', '!']
```

12.. Write a Python program to check if all items in a given list of strings are equal to a given string.

```
string_list = ["apple", "apple", "apple"]
target_string = "apple"
all_equal = all(item == target_string for item in string_list)
print("All items are equal to the target string:", all_equal)
```

```
All items are equal to the target string: True
```

Gateway Classes

PYTHON PROGRAMMING

UNIT 2

Python Program Flow Control Conditional blocks

By

Pragya rajvanshi

LEC-6(some mixed questions)

Gateway classes

1. **Write a Python program to replace the last element in a list with another list. Sample data : [1, 3, 5, 7, 9, 10], [2, 4, 6, 8] Expected Output: [1, 3, 5, 7, 9, 2, 4, 6, 8]**

```
list1 = [1, 3, 5, 7, 9, 10]
list2 = [2, 4, 6, 8]
result = list1[:-1] + list2
print("Expected Output:", result)
```

```
Expected Output: [1, 3, 5, 7, 9, 2, 4, 6, 8]
```

2. Write a Python program to check whether the n-th element exists in a given list.

```
sample_list = [10, 20, 30, 40, 50]
n = 3
if n < len(sample_list):
    print(f"The {n}-th element exists: {sample_list[n]}")
else:
    print(f"The {n}-th element does not exist in the list.")
```

```
The 3-th element exists: 40
```

3. Write a program to create a list of empty dictionary

```
n = 5
empty_dict_list = [{} for _ in range(n)]
print("List of empty dictionaries:", empty_dict_list)
```

```
List of empty dictionaries: [{}, {}, {}, {}, {}]
```

Gateway classes

4. Write a Python program to reverse a given list of lists

```
input_list = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
reversed_list = [sublist[::-1] for sublist in input_list[::-1]]
print(reversed_list)
```

```
[[9, 8, 7], [6, 5, 4], [3, 2, 1]]
```

5. Write a Python program to remove specific words from a given list.

Original list:
['red', 'green', 'blue', 'white', 'black', 'orange']
Remove words:
['white', 'orange']
After removing the specified words from the said list:
['red', 'green', 'blue', 'black']

```
['red', 'green', 'blue', 'black']
```

```
original_list = ['red', 'green', 'blue', 'white', 'black', 'orange']
remove_words = ['white', 'orange']
filtered_list = [word for word in original_list if word not in remove_words]
print(filtered_list)
```

Gateway classes

6 . Write a Python program to combine two lists into another list randomly.

```
import random
list1 = [1, 2, 3, 4, 5]
list2 = ['a', 'b', 'c', 'd', 'e']
combined_list = list1 + list2
random.shuffle(combined_list)
print(combined_list)
```

```
['c', 'd', 'e', 5, 2, 3, 4, 'b', 'a', 1]
```

Gateway classes

Difference between while and for loop

For loop	While loop
For loop is used to iterate over a sequence of items.	While loop is used to repeatedly execute a block of statements while a condition is true.
For loops are designed for iterating over a sequence of items. Eg. list, tuple, etc.	While loop is used when the number of iterations is not known in advance or when we want to repeat a block of code until a certain condition is met.
For loop require a sequence to iterate over.	While the loop requires an initial condition that is tested at the beginning of the loop.
For loop is typically used for iterating over a fixed sequence of items	While loop is used for more complex control flow situations.
For loop is more efficient than a while loop when iterating over sequences, since the number of iterations is predetermined and the loop can be optimized accordingly.	While a loop may be more efficient in certain situations where the condition being tested can be evaluated quickly.

Gateway Classes

7.write a program to check whether entered number is Fibonacci number or not

```
1  import math
2  def isPerfectSquare(x):
3      s = int(math.sqrt(x))
4      return s*s == x
5  def isFibonacci(n):
6      return isPerfectSquare(5*n*n + 4) or isPerfectSquare(5*n*n - 4)
7  i=int(input("enter the number"))
8  if (isFibonacci(i) == True):
9      print(i,"is a Fibonacci Number")
10 else:
11     print(i,"is a not Fibonacci Number ")
12
```

input

```
enter the number3
3 is a Fibonacci Number
```

Gateway classes

8.write a program to check whether entered number is Fibonacci number or not in range

```python
import math
def isPerfectSquare(x):
    s = int(math.sqrt(x))
    return s*s == x
def isFibonacci(n):
    return isPerfectSquare(5*n*n + 4) or isPerfectSquare(5*n*n - 4)
a=int(input("enter the terms"))
for i in range(1,a):
    if (isFibonacci(i) == True):
        print( i,"is a Fibonacci Number")
    else:
        print( i,"is a not Fibonacci Number ")

```

input

```
enter the terms6
1 is a Fibonacci Number
2 is a Fibonacci Number
3 is a Fibonacci Number
4 is a not Fibonacci Number
5 is a Fibonacci Number
```

Oday classes

Use of else with loops

Yes, you can use else with loop

```
l = [1, 2, 3, 4, 5]
for a in l:
    print(a)
else:
    print("else block is executed")
```

```
1
2
3
4
5
else block is executed
```

```
1  i=0
2  while i<5:
3      i+=1
4      print("i =",i)
5  else:
6      print("else block is executed")
```

```
i = 1
i = 2
i = 3
i = 4
i = 5
else block is executed
```

Gateway classes

Conditional statement:

- If statement
- If-else statement
- Nested-if else
- Elif statement

Purpose of loops

- Loops are used in programming to execute a set of instructions repeatedly until a certain condition is met. They help automate repetitive tasks, reducing the amount of code needed and improving efficiency

Gateway classes

AKTU QUESTIONS		
Q.1	What are conditional statement in python using small code example	AKTU 2019-20
Q.2	Write a python program to check if the input year is a leap year or not	AKTU 2021-22
Q.3	Write a program to produce the Fibonacci series in python	AKTU 2021-22
Q.4	Define loops. Explain different types of loops in python Or Difference between while and for loop with example, syntax and flow diagram	AKTU 2023-24
Q.5	ANY pattern question	AKTU 2021-22 AKTU 2022-23

Gateway classes

AKTU QUESTIONS		
Q.6	Explain the the working and purpose of loops. Discuss break and continue with example .Write a python program to convert time from 12 hour format to 24 hour format	AKTU 2019-20
Q.7	Explain the use of break , pass and continue with the suitable example	AKTU 2021-22
Q.8	Write a program to check an input number is prime or not	AKTU 2021-22
Q.9	Write a program to calculate the reverse of a number	AKTU 2022-23
Q.10	Write a program to check whether a number is Fibonacci number or not	AKTU 2019-20
Q.11	Write a program to convert time format from 12 hour to 24 hour format	AKTU 2019-20

Gateway classes



Gateway Classes